Abstract

With this paper, we propose a matchmaking system that performs matchmaking of requested grid services with that of advertised ones to discover the best suitable services. The matchmaking algorithm implemented in our matchmaking system considers functionality of the requested service as critical factor for faster and accurate discovery of services. Unlike conventional matchmaking algorithm where service discovery is based on Inputs and Outputs, our algorithm retrieves services similar to the one requested based on functionality in addition to Inputs and Outputs.

1. Introduction

The semantic Grid is characterized by an open system, with a high degree of automation, which supports flexible collaboration and computation on a global scale [1, 2]. In such an environment it is essential to provide facilities to the user for easier discovery of the services. The concept of matchmaking plays a vital role in semantic discovery of services that matches together the features requested by the requester with that of advertised ones. The conventional matchmaking algorithm compares Inputs and Outputs of the advertised service with that of the requested ones [3]. In this conventional approach, the algorithm can also retrieve irrelevant services just because of their Inputs and Outputs are matching with that of requested service.

We address this issue by proposing a Matchmaking system for discovering grid services semantically in this research work. A novel Matchmaking algorithm is introduced and implemented in which, the system takes functionality of the requested service into account prior to the comparison of inputs and outputs. This research work has been carried out as a part of Garuda project, a first national initiative in India to provide a test bed for research and engineering of technologies, architectures, standards and applications in Grid Computing [4].

2. Matchmaking System

The architecture of the proposed matchmaking system is shown in the Figure 1. It identifies necessary components needed for creating service advertisements and discovery. The service provider describes the characteristics of the service in terms of Web Service Description Language (WSDL) file. This WSDL file is then converted into OWLS (Web Ontology Language for Services) description using WSDL2OWLS tool of OWLS editor [5]. The OWLS description describes the grid services semantically which can be queried using any of the available inference engine, for example Algernon. The information retrieved from the advertised OWLS description will be used to match with that of the requested requirements. The matchmaking system compares the functionality, outputs and inputs (FOI) of the requested service with that of advertised ones. To determine the semantic similarity between the requested FOI and advertised ones are determined using the concept of ontology. Web Ontology Language (OWL) makes it possible for concepts to be described and thereby complex concepts can be built up in definitions out of simpler concepts [6]. Protégé, an OWL editor developed by Stanford University, facilitates the
creation of ontology. An inference engine can be used to determine the semantic similarity between two different concepts of ontology by using appropriate reasoner axioms.

In this paper, Algernon inference engine [8] is used to interact with ontology knowledge base for semantic retrieval of information. Also, we build function ontology to determine semantic similarity between functionality of the requested and that of advertised. Similarly, domain ontology has been built for determining inputs and outputs similarity. The proposed algorithm determines the degree of similarity between the advertised FOIs and the requested ones to perform matchmaking. Based on the similarity, the algorithm infers subsume relation if the requested concept is the subconcept of the advertised concept. As result of functionality comparison, the algorithm retrieves a set of advertised service descriptions along with respective degrees of match that is exact or subsume. In the successive stages the degree of closeness between the outputs and inputs are determined by referring domain ontology and irrelevant services are filtered out at every stage. By incorporating functionality matching in the initial stage, services performing relevant functions are only selected by the algorithm for further comparison. This will greatly improve the precision of the results [11].

3. Concept of Matchmaking System

In our context, matchmaking refers to capability matching of the requested services with the advertised services. Also, we assume capability of a service can be expressed in terms of functionality that the service offers, the kind of input it takes and output it produces. In our algorithm, we consider these three parameters of a service for matchmaking process. Further, with domain and function ontology in the background, it is possible to determine semantic similarity of every parameter and computes different degrees of match. Though, the literature specifies five different degrees of match [9] viz., exact, subsume, plugin, intersection and disjoint, we ignore plugin and intersection matches. This is because, they both do not represent similar type of services [10].

3.1. Matchmaking Algorithm

The algorithm starts from extracting FOI from each of the service advertisement descriptions. Similarly, FOI of the requested service is obtained from the service requester. Since, the functionalities are modeled as concepts and subconcepts in the function ontology, the semantic relation between the advertised and requested concepts are determined using suitable algernon queries which can be Exact or Subsume or Disjoint. The algorithm infers subsume relation if the requested concept is the subconcept of the advertised concept.

As result of functionality comparison, the algorithm retrieves a set of advertised service descriptions along with respective degrees of match that is exact or subsume. In the successive stages the degree of closeness between the outputs and inputs are determined by referring domain ontology and irrelevant services are filtered out at every stage. By incorporating functionality matching in the initial stage, services performing relevant functions are only selected by the algorithm for further comparison. This will greatly improve the precision of the results [11].

Also, the time taken to discover suitable services is also reduced as major number of advertised services with irrelevant functionalities is filtered out.

4. Implementation

The algorithm is implemented in three sequential phases. In every phase, we eliminate irrelevant advertised services by comparing their FOI with that of the requested ones. Suitable algernon axioms are written.
to interact with function and domain ontology and the
degree of closeness between FOI of advertised and
requested ones are inferred. The advertised services for
which the degree of closeness is either exact or
subsume is considered for next phase of comparison.

<table>
<thead>
<tr>
<th>Requested Service Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service.Name</td>
<td>Mathematical Service</td>
</tr>
<tr>
<td>Service.InputsNumber</td>
<td>2</td>
</tr>
<tr>
<td>Service.Inputtype</td>
<td>float, int</td>
</tr>
<tr>
<td>Service.Output</td>
<td>1</td>
</tr>
<tr>
<td>Service.Outputtype</td>
<td>int</td>
</tr>
<tr>
<td>Service.FunctionalityNumber</td>
<td>2</td>
</tr>
<tr>
<td>Service.Functionality</td>
<td>Add, Sub</td>
</tr>
</tbody>
</table>

Figure 2: Service Request Description

For illustration purpose, we consider the service request description shown in the figure 2. Also, we consider a simple advertised service that performs various arithmetic operations viz., addition, subtraction, multiplication and division. Also, we assume that these functions produce an integer output and accept two inputs of type double and int. The service description file of this service is converted into corresponding OWLS descriptions using WSDL2OWLS tool. The functionality, outputs and inputs are extracted from the OWLS descriptions using suitable Algernon axiom. In the initial phase of the algorithm, the functionality of the advertised and requested are compared. In our case, the requested functionality is add and sub whereas the advertised are addition, subtraction, multiplication and division. It means that the advertised functionalities subsume the requested functionalities and this advertised service will satisfy the request. In the second phase, we determine the semantic similarity of the outputs between the advertised and requested. Here, both the requested and advertised output type is “int”. While referring to the domain ontology, it is found that both the output types are matching exactly. However, in the final phase, while comparing the inputs, it is found that only one of the requested inputs is exactly matching with that of the advertised ones. To determine the semantic similarity between the other input variable, the domain ontology is referred. In our domain ontology, the “float” type is modeled as sub concept of “double”. It means that “double” type offers more capability than that of “float” which is requested. Hence, the algorithm concludes that there exists a “subsume” relationship between the advertised and requested inputs.

Once the algorithm completes the three stages, it delivers a set of services to the requester in which irrelevant set of advertised services are eliminated completely.

5. Experimental Results

We evaluate the performance of our matchmaking algorithm using various inputs for different number of services.

For case study, we have considered twenty grid services with seven different types of functionalities. For every service request, we calculated the time taken by our algorithm and also the time taken by conventional algorithm to discover services. By observing the results as shown in the figure 3, we found that the performance of our algorithm is better than that of conventional method as it takes less time to discover services. This is because, the functionality based algorithm eliminates handful of advertised irrelevant services before comparing their output and inputs with the requested ones.

We also estimate the efficiency of our algorithm in terms of irrelevance_factor as efficiency depends on retrieving the correct results as well as eliminating the unwanted ones. We can define irrelevance_factor as follows:-

\[
\text{Irrelevance_factor} = \frac{|\text{servicesfound} - |\text{relevantservices}|}{|\text{servicesfound}|}
\]
Two sets of results have been obtained by considering nine and twenty sets of advertised services. We calculated irrelevance_factor for every set of input request for different number of services obtained from the conventional algorithm. The results are plotted for both set of advertised services as shown in Figure 4 and they are compared against results from FOI algorithm. We observed that on an average, the conventional matchmaking algorithm gives an irrelevance factor of 65%. Since our functionality based matchmaking algorithm eliminates all irrelevant services, an irrelevance factor of 0% is obtained thereby showing better efficiency of the system.

6. Conclusion

With this paper, we address the problem of semantic discovery of grid services by proposing a matchmaking system. It performs semantic matching of requested grid services against the advertised ones. The matchmaking algorithm implemented in this paper considers functionality as critical factor and retrieves services with similar capabilities. Hence, the results obtained by our algorithm are proven to be more accurate with greater precision and efficiency.

Acknowledgements

This work was undertaken as part of “Garuda: The National Grid Computing Initiative”, India and is financially supported by the Centre for Development of Advanced Computing, India.

7. References

8. algernon-j.sourceforge.net/doc/commands/
2003.